

Tutorial V.I - Programming with AI

Programming: Everyday Decision-Making Algorithms

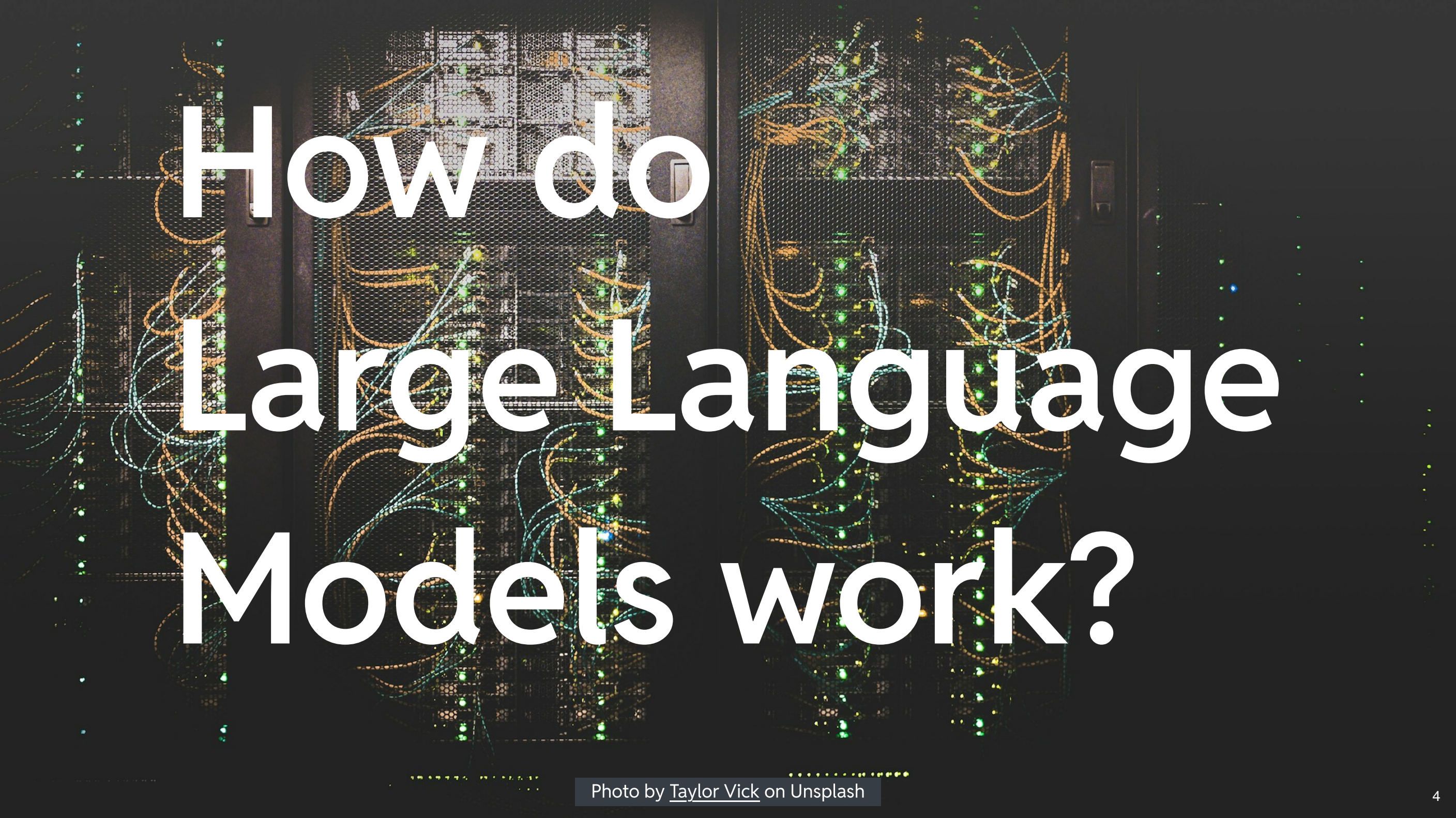
Dr. Tobias Vlček

Kühne Logistics University Hamburg - Winter 2025

Programming with AI

Using AI to generate code

- Coding by hand is **not the only way to generate code**
- Most likely, a lot of you have already used **ChatGPT**



How do Large Language Models work?

Photo by [Taylor Vick](#) on Unsplash

Large Language Models (LLMs)

- Think of them like **advanced pattern recognition systems**
- They have “read” **massive amounts of text**
- Books, websites, articles, code, and more
- Text is broken into **tokens**, parts of words or punctuation
- Based on patterns, they can **generate new text**

Training LLMs

- Imagine learning a language by **reading millions of books**
- Learns patterns in **how words and ideas connect** via tokens
- Interconnected nodes with **weights representing patterns**
- During training, these **weights are adjusted**
- Once trained, **applying** them takes much less resources



Using a trained model is called **inference**.

Pattern Recognition

- **Not like a search engine!**
- When asked, it looks for **relevant patterns** it learned
- Like having a **huge library** in its “memory” to draw from
- It can find **patterns between concepts** and your question
- Knows only limited text at once (**context window**)



Managing **context windows** is crucial!

Probability based responses

- After each written token, it predicts **“what should come next?”**
- Like a advanced version of the **word prediction** on your phone
- Chooses the **most likely next token** based on training
- **But can't actually “think” or “understand” like humans**

Limitations

- No true understanding of cause and effect¹
- Sometimes makes mistakes or “hallucinates”
- Mostly only knows what it was trained on² and can reflect biases
- No emotional understanding (but **can simulate responses!**)³

Impact on Jobs

- **Question:** What do you think about their impact on jobs?
- **Question:** What are the implications for us?
- **Question:** Can we use them to our advantage?



If you use free models, be aware that your prompts are going to be used by the providers and are not private. But for learning and experimenting, this should be no issue.

A Great Overview by 3Blue1Brown

- Greg Sanderson provides an excellent explanation of LLMs
- Great starting point to understand LLMs
- Check out his [YouTube channel, 3Blue1Brown](#) for more

AI Coding Partner

What is GitHub Copilot?

GitHub Copilot is an *AI pair programmer* that helps you write code faster and with less effort.

Think of it as:

- An autocomplete for entire lines or blocks of code
- A coding assistant that understands context
- A learning tool that shows you coding patterns

 Copilot uses AI trained on billions of lines of public code to suggest completions.

 The same idea ships in many editors (Zed, Cursor, VS Code with various extensions): an assistant that reads your code and suggests the next step.

Used autocomplete before?

When you type on your phone, it suggests the next word.

GitHub Copilot does the same for code:

- You write a comment describing what you want
- Copilot suggests the code to do it
- You accept, modify, or reject the suggestion

It's autocomplete but much better than what you are used to

Why Use Copilot?

Benefits you while learning and working with Python:

- **Faster coding:** Less time typing boilerplate code
- **Learn patterns:** See how experienced programmers code
- **Understand syntax:** Get correct syntax without memorizing
- **Stay in flow:** Focus on logic, not syntax errors



Especially helpful when you know **WHAT** you want to do but forget **HOW** to do it.

When should you use Copilot?

Good uses of Copilot

- Understanding Python syntax you forgot
- Writing repetitive or boilerplate code
- Getting unstuck on simple problems
- Exploring different approaches

What shouldn't you do with Copilot?

Not so good uses of Copilot

- Replacing learning fundamentals
- Accepting code you don't understand
- Skipping practice exercises
- Copy-pasting without reading

Always understand what Copilot suggests before accepting!

But of course I know you will not do that ;)

Accept without reading?

Copilot might suggest code that:

- Works but uses concepts you haven't learned yet
- Contains subtle bugs or edge cases
- Doesn't match your specific requirements
- Uses inefficient approaches
- Introduce dangerous code in your project



Dangerous code can lead to security vulnerabilities, data loss, or other issues. In the context of this lecture it should be no issue, but in companies it can be one!

My take: Just be careful, ok?

Your code, your responsibility:

1. **Read** the suggestion carefully
2. **Understand** what it does
3. **Test** it with examples
4. **Modify** if needed



Don't accept code blindly, especially later if things are more complicated.

Getting Started

Get Free Access



GitHub Copilot's free student signups are currently paused, so we use one of these instead. Everything this lecture shows about AI pair programming works the same way.

Option 1 — Mistral (free “Experiment” tier):

1. Go to console.mistral.ai and sign up (phone verification, no credit card)
2. In the **Admin Console** → **Privacy** menu, disable “Anonymous improvement data” so your inputs are not used for training
3. Chat at chat.mistral.ai or connect the API key to your editor

Get Free Access II

Option 2 — Zed's built-in AI:

1. Install Zed and sign in with a GitHub account
2. Apply for the free student plan at zed.dev/education (12 months of model credits)

Verifying It Works

1. Create a new Python file (`.py`) and write a comment describing a small function
2. Ask the assistant to complete it (inline suggestion or chat)

You should see:

- A code suggestion appearing next to or below your comment
- Accept it, then **read it line by line** before running



Try to get one of the two options running on your own until next session.

Summary

Key Takeaways

1. **GitHub Copilot is a tool:** You still need to learn fundamentals
2. **Understand before accepting:** Read every suggestion carefully
3. **Use it strategically:** Syntax help yes, thinking replacement no
4. **Get free access:** Mistral's free tier or Zed's student plan
5. **Practice:** The more you use it, the more helpful it becomes